An investigation of two real time machine learning techniques that could enhance the adaptability of game AI agents

David King and Cassie Bennett

This paper presented at GameOn'2016 17th International Conference on Intelligent Games and Simulation, Lisbon, Portugal, 13-15 September 2016

ISBN 9789077381946

AN INVESTIGATION OF TWO REAL TIME MACHINE LEARNING TECHNIQUES THAT COULD ENHANCE THE ADAPTABILITY OF GAME AI AGENTS

David King
Cassie Bennett
Abertay University
40 Bell Street, Dundee, United Kingdom, DD1 1HG
Email: d.king@abertay.ac.uk

KEYWORDS

Artificial Intelligence (AI), Adaptive Game AI, *Q*-Learning, *N*-Gram Prediction.

ABSTRACT

Developers strive to create innovative Artificial Intelligence (AI) behaviour in their games as a key selling point. Machine Learning is an area of AI that looks at how applications and agents can be programmed to learn their own behaviour without the need to manually design and implement each aspect of it. Machine learning methods have been utilised infrequently within games and are usually trained to learn offline before the game is released to the players. In order to investigate new ways AI could be applied innovatively to games it is wise to explore how machine learning methods could be utilised in real-time as the game is played, so as to allow AI agents to learn directly from the player or their environment. Two machine learning methods were implemented into a simple 2D Fighter test game to allow the agents to fully showcase their learned behaviour as the game is played. The methods chosen were: *Q*-Learning and an *N*-Gram based system. It was found that *N*-Grams and *Q*-Learning could significantly benefit game developers as they facilitate fast, realistic learning at run-time.

INTRODUCTION

There are a wide range of characteristics that can be used to categorise how intelligence can be represented within computer programs. Definitions of intelligence include the ability to make a decision based on information that has been obtained from the world or the ability to solve problems. Others would argue that for something to be recognised as intelligent, it must be able to exhibit evidence of learning and adaptation (Bourg and Seemann 2004a), something which has rarely been seen in games before. Agents that are able to constantly adapt could completely change the landscape when applying AI within games. Therefore, when considering how games should evolve in the future, it is wise to take into account AI that learns and directly reacts specifically to each player.

The opportunity for increasingly complex AI techniques in games is improving as computational power in consoles and computers evolve (Bourg and Seemann 2004b; Vasquez II 2011). Recently, the games industry has been heavily focused on improving the graphical quality of games, however AI is now one of the main elements of a game that allows it to stand out and make a real impact on the market. Unique, interesting, and impressive AI is becoming the main attraction of games (Schwab 2009). In particular, AI learning methods and the use of machine learning techniques within games during run-time is a largely unexplored territory in game development, but a popular field of research for academic uses (Dill 2011). There is a wealth of potential in applying machine learning techniques to games, as this could lead to having AI agents that adapt their behaviour to the current player and give a unique, personalised experience. Utilising learning techniques would allow AI agents to give unique reactive behaviour in response to individual players, which in turn could provide the distinctive breakthrough a game needs to give it a competitive edge. In addition, this would combat the problem of interactions with Non-Playable Characters (NPCs) becoming boring and predictable as a game goes on, which regularly leaves room for exploitation of the NPC behaviour and actively diminishes the challenge of the game (Bourg and Seemann 2004a).

It is extremely rare but not unheard of for games to utilise machine learning methods at run-time. NERO (NeuroEvolving Robotic Operatives) is a game that allows players to use Artificial Neural Networks (ANN) to train agents to fight other NPC agents (NERO Team [no date]). However, it would be beneficial to investigate how behaviour could be adapted when the AI is learning from the players own behaviour during a game.

These learning techniques could provide agents with completely tailored behaviour and reactions towards players. There is a possibility that AI agents learning from player behaviour could be detrimental to the gameplay, but on the other hand it could open up so many opportunities for different types of games and even the possibility of unique games that will stand out in a competitive market. Not only that, but using these techniques could increase the shelf-life of a game due to the many different ways to play it that this would provide (Stanley et al. 2005).

The focus of this paper is the utilisation of different AI learning methods that will allow AI agents to adapt to individual players' playing styles as the game runs in real-time. The paper aims to record the process and evaluation of developing, designing and comparing two different machine learning techniques in order to present methods that are well suited, and can be realistically implemented, within games. The overall aim is to investigate if, and how, implementation of agents that learn can give each player a unique, tailored experience when interacting with them.

Machine Learning

There are several methods that facilitate learning, and the choice of which to use is largely application dependent.

Supervised learning is a technique often used for backpropagation in Neural Networks and Decision Trees (Mathworks [No date]). Supervised learning uses training data provided to it in order to adjust internal parameters (such as weights) to provide the desired output. This technique is useful only if the desired output for the training data is known.

Unsupervised learning is a technique commonly used in methods such as Self Organising Map Neural Networks (SOM), Adaptive Resonance Theory Neural Networks (ART), clustering algorithms such as *K*-Means and predictive techniques like *N*-grams. AI systems use this technique to categorise data by independently observing, and finding patterns or similarities in the inputs (AI Horizon [no date]).

Reinforcement learning allows the AI agent to autonomously learn through experiencing the world, obtaining rewards or punishments given in response to their actions, which then influences their future decisions. Examples of Reinforcement Learning include *Q*-Learning, SARSA, and Temporal Difference Learning. The goal of the AI agent is to try different actions in order to make decisions based on which one gives them the largest reward (Whiteson 2007).

Supervised learning is less suited to games than reinforcement learning as it requires a human expert to determine the desired outputs for the agent, and this limits the ability to learn during the course of a game (Whiteson 2007). Reinforcement agents learn as they independently gain more experience from the world and do not require a human to guide their behaviour, allowing real-time learning without the need for human intervention.

METHODOLOGY

With a wide selection of learning models to choose from, this paper looks at two in detail that each have distinctive approaches to learning, are relatively easy to apply and are therefore appropriate for real-time applications; Reinforcement Learning as utilised in *Q*-Learning, and Unsupervised Learning as seen in *N*-Gram prediction.

Reinforcement Learning

Q-Learning was developed by Christopher Watkins in 1989 (Watkins and Dayan 1992), and relies on experience based knowledge to focus on making optimal decisions based upon the outcome of interactions in the world (Poole and Mackworth [no date]). It is a type of reinforcement learning for AI agents that uses trial and error to learn more about the world, actions, and consequences. AI agents carry out actions, and based on the outcome they are given a value as a 'reward' or 'punishment' so that the agent can record this and try to make a more optimal decision next time (Watkins and Dayan 1992).The agents check and update the *Q*-Value, which is a function of the current state and the chosen action, based on the experience they gain as they continually attempt to solve specific problems. The *Q*-Value is increased if the agent is rewarded in order to improve the probability of the agent choosing that action again when in the same state, whereas for punishment the *Q*-Value is reduced to make it less likely that it will be chosen (DeWolf 2012). The agent eventually learns the optimal policy by recurrently attempting actions in each state and finding the best *Q*-Value for that particular action-state pair (Poole and Mackworth. [No date].

Q-Learning has four parts for every decision: The initial state, the action taken, the reward, and the new state to which the agent has moved. Each action can be represented by this sequence and the agent's knowledge of the game space only changes when it carries out an action and lands in a new state. This means the agent learns from its interactions and the consequences it experiences in order to improve and make better decisions. An example of this method being used are AI programs that can learn how to play video games, such as Google's Deep *Q*-Network program (Lewis 2015), however this could be integrated into agents within games in order to learn from the player.

The *Q*-Value that represents how effective an action is in a given state is calculated using an iterative process in order to refine the *Q*-Value estimate (Poole 2010) as shown in Equation (1) below:

$$Q_{n+1}(S,A) = (1-L)Q_n + (LR) \qquad (1)$$

where *S* is the current state, *A* is the action chosen, *L* the learning rate and *R* the reward value. The above rule uses the reward given along with the learning rate in order to determine the new *Q*-Value. The learning rate is a value between 0 and 1 that determines how much affect the current *Q*-Value has on the newly calculated *Q*-Value. The larger the learning rate, the more influence the reward has on the new *Q*-Value, and the less effect the current *Q*-Value has. For the test game, tuning the learning rate to produce the best outcome resulted in a value of 0.5. This learning rate was suitable as the *Q*-Value relied equally on both the current *Q*-Value and the reward, which allowed the agent to learn quickly as well as reliably. Reward values are dependent on the result of the action, so as to determine the appropriate value that will encourage or deter an action from happening again. For this application the reward values given for the various actions are reliant on the health changes of the player and agent, and are shown in Table 1.

Reward values were designed to appropriately encourage the agent to learn from its mistakes, and to aim towards higher rewarded actions throughout the game. The *Q*-Value is calculated each time the player performs an action, so that the Agent counters this with the action given by the *Q*-Learning algorithm. The action with the highest quality value in that state is chosen when the action is being determined, which ensures that the agent is performing the most desired action in retaliation.

Unsupervised Learning

N-Grams are a type of unsupervised learning technique used in order to learn patterns in sequences. Through the use of string matching, the current actions of the player are compared to a record of the previous sequences of actions to find identical sequences for prediction (Tucci 2014). Sequences are stored in a window of size *N* to be checked. For example a 4-Gram records the frequency of a sequence of four actions, and when the player next performs the first three actions the fourth is predicted (Millington and Funge 2009). When predicting, the most frequent action that follows a sequence of the player's current actions up to a window of size $N - 1$ is chosen. It is important however that the size of the window is suitable for the range of actions available to the player. If the window size is too small predictions will be less accurate as there is not enough

history to check, whereas if the window size is too big predictions will be less accurate due to randomness in the history and sequences are less likely to be matched to an *N*-Gram (Tucci 2014).

Table 1: Reward Table

Result	Reward Value	Reason
Both Player and Agent were not hit	0	No reward given for no effect on Player or Agent
Player and Agent damaged each other	0.3	Small reward given because damaging the Player is a positive action, however not given full positive reward because Agent was damaged also.
Player damaged, Agent safe	1.0	Largest reward given because damage caused to player, but Agent took no damage
Agent damaged, Player safe	-1.0	Largest negative reward given, because Agent took damage while Player did not
Agent hit, but health did not change	0.7	Large positive reward, as this means the Agent blocked the attack correctly
Player hit, but health did not change	-0.5	Negative reward, as this indicates the player blocked the attack successfully

This technique is sometimes used in combat/fighting games, as it finds patterns in the input or sequence of events by looking at their history as they happen, and can therefore react specifically to the player's current action (Millington and Funge 2009). This means a co-operative AI agent could imitate the player's style to benefit the player in game play, or an enemy AI agent can adapt its style uniquely against each player. It is a type of learning algorithm that would lend itself well to games where the player has a specific style they use for game play, because the *N*-Grams could then use the player's input history to learn their patterns and hence adapt to the player (Vasquez II 2011).

Developing the Test Game

The design and creation of the test game was heavily focused on what kind of game would provide instant, realistic, and clear learning abilities in AI agents if machine learning was used to control their behaviour. When reflecting on the criteria needed for the game, a 2D Fighter game with an AI controlled opponent, similar to Street Fighter (Capcom 1987) or Mortal Kombat (Midway Games 1992), was chosen.
The player interacts with the game by pressing controls that correspond to moves the player can make. Both the player and the AI agents can move around the screen in order to get close enough to attack the other, or to move out of their range. A screenshot of the game while it is running is shown in Figure 1. The moves that the AI agent and player can perform are: jump, crouch, move, punch, low kick, high kick, low block, and high block. The advantage of demonstrating the agent's learning capabilities in this type of game is that it is clear to see how the agent's knowledge improves over time. For example, the player might punch the agent, and the agent will take damage. All the agent will know is that its upper body was hit, and it lost health. As its knowledge improves and it tries different moves in response to this, the agent should learn strategies such as blocking its upper body when the player punches, or punching the player back. This type of reaction shows the player how the agent has been learning from its experience during the fight. Based on the effect of the players move on the AI agent, the agent can learn to predict or counter those moves more effectively as the fight progresses.

Figure 1: Screenshot from the Game

EVALUATION

Qualitative Evaluation

In order to evaluate the machine learning methods implemented in the test game a questionnaire was developed with the aim of gathering information on their effectiveness. The questionnaire was created using Google Forms that testers could fill in online, and the game's test build was distributed via a download link on Google Drive.

Quantitative Evaluation

The quantitative testing focused on the technical side of implementing the machine learning techniques. Technical qualities to be tested were inspired by the prominent computer scientist Pieter Spronck's list of requirements for successful online learning algorithms. In the paper 'Online Adaptation of Game Opponent AI in Simulation and In Practice,' Spronck et al. state that online learning methods must be "Fast, effective, robust, and efficient" in order to be successful in a real time environment (Spronck et al. 2003). Therefore, the following aspects of each method were evaluated:

- Processing speed during run-time
  (Evaluation of speed and efficiency)

- Accuracy and error
  (Evaluation of effectiveness and robustness)

RESULTS

Qualitative Results

Twelve testers participated in playing the game and completed the questionnaire, which was split up into three sections:

- Section 1: *Q*-Learning (6 questions)
- Section 2: *N*-Grams (6 questions)
- Section 3: Comparison (12 questions)

Testers were asked to indicate their thoughts on the behaviour of the AI Agent using a Likert scale from 1 to 5; where 1 meant did not agree at all and 5 meant they agreed extremely. In addition, several other questions were posed to gain more insight into the testers' decisions, the results of which are included below. The following aspects were examined:

Realistic: It was important to ask for the testers' opinions on how realistically the agent behaved, as AI in games needs to be highly believable in order to be immersive, whereas unrealistic AI agents can discourage players by frustrating them.

Intelligent: The AI agent needed to act or give the illusion of intelligence to the player, so that the agent's decision making seemed logical and understandable and thus prevented the player from losing immersion in the game.

Reactive: All players take their own approach when playing games, therefore in order to be truly adaptive the agent had to feel as though it reacted to the player's own method of playing.

Interesting: The AI agent needed to be interesting to the player. If it exhibited boring behaviour, this would lose the players attention quickly and would not entice them to play the game.

Enjoyable: Lastly, the agent's behaviour needed to provide enjoyable behaviour to the player as enjoyment is the primary focus of video games. If the enjoyment of a game is increased by using learning algorithms for agents, this would be a clear sign that adaptable AI in games would be beneficial for future games.

Figure 2 shows the results of the question evaluating the *Q*-Learning agent, whilst Figure 3 shows the results for the *N*-Gram agent.

Realism

In answer to the question "Which method did you find to be the least realistic?" the results were 50-50. So there was no overall preference for either method

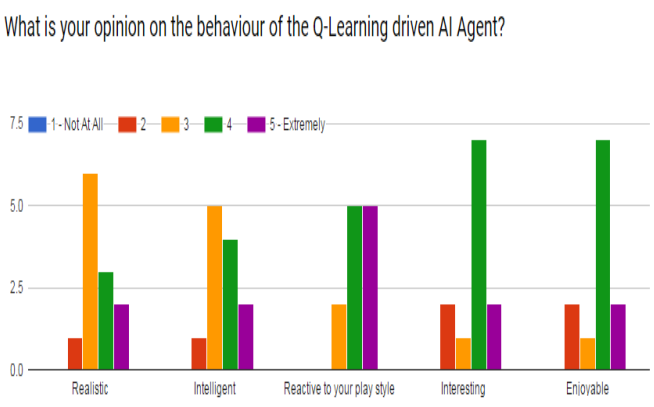

Figure 2: Results for *Q*-Learning Agent Attributes

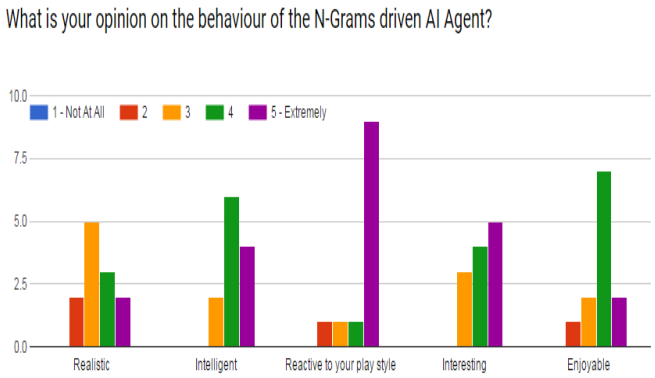

Figure 3: Results for *N*-Gram Agent Attributes

Intelligence

Each tester was asked to state to what extent they felt that the agent displayed evidence of learning while playing the game. The results are compared in Figure 4.

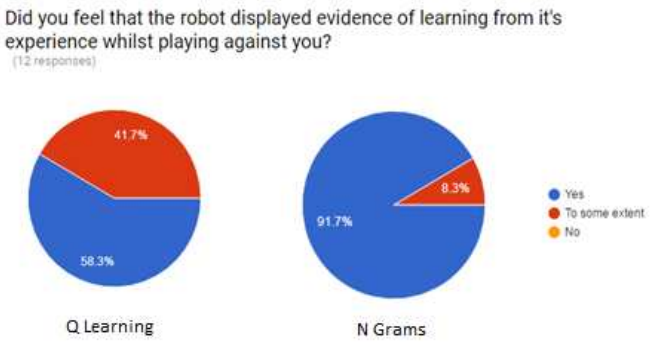

Figure 4: Pie Charts Displaying How Testers Felt About the Intelligence of the AI Agents

The polls found that on average, *Q*-Learning's intelligence value was 3.58 out of 5, whereas *N*-Grams value was 4.16. Furthermore, looking at the charts in Figure 4 it is evident that testers felt the robot utilising *N*-Gram based learning was significantly more intelligent than the *Q*-Learning robot; 97% stated that the *N*-Gram robot displayed evidence of learning and only 58.3% stated the same for the *Q*-Learning robot. Expanding on their choices, testers explained that *N*-Grams exhibited learning more clearly because it learned how they played the game, and testers had to change their own tactics in order to defeat the robot. For *Q*-Learning, some felt that the robot did exhibit intelligence clearly as it learned from its mistakes and began to block, attack and counter appropriately against the tester's actions. However, some felt that *Q*-Learning would have been able to show better intelligence if the agent had a longer time to learn because it did not learn as fast as the *N*-Gram agent. On the other hand, when fighting the *N*-Gram agent testers found that it learned so quickly that they had to try and outsmart the agent during the fight as it soon became difficult.

What is interesting is that two players noted behaviours that they believed displayed the *N*-Gram agent's intelligence which were not actually true. These testers noted that the *N*-Gram robot would 'change its tactics,' and 'employ tactics,' to sabotage the players fighting style. This is fascinating because artificial intelligence in games is largely just an illusion, as players make connections in their head as to how the AI agents are thinking based on what they observe. The *N*-Gram agent does not change its tactics throughout the game, and merely gains more information about the player to make more precise predictions, however the behaviour

exhibited by the agent gave players a stronger illusion of intelligence.

Reactivity

Testing the reactivity of the agents was important as one of the main aims of the project is to explore how machine learning methods can enhance this aspect of game agents. In the poll, the *N*-Gram agent again held a higher average value for its reactivity, with its value being 4.5 and *Q*-Learning's value being 4.25. To explore this further, the testers were asked to select which agent they felt was the most reactive, the result of which is shown in Figure 5.

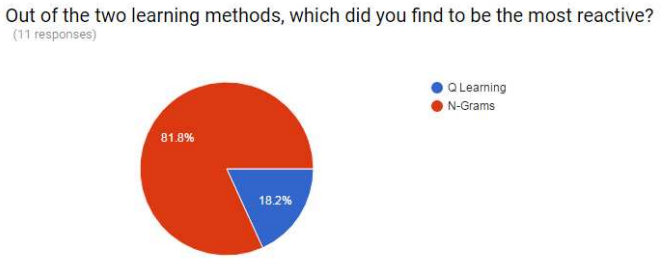

Figure 5: Pie Chart Displaying How Testers Felt About the Reactivity of the AI Agents

As shown in the pie chart, a substantial amount of testers chose the *N*-Gram method to be more reactive than *Q*-Learning. This highly suggests that *N*-Grams is well suited for quick learning in games during real time, as testers easily identified this as the most reactive method to play against. Testers who chose *Q*-Learning for this question noted that they believed the *Q*-Learning robot was more prepared for their actions than the *N*-Gram. In addition, they felt that it learned to react quickly and was reactive to their individual play style. However once they identified a technique they could use, the robot became too easy to defeat. Testers who selected *N*-Grams as most reactive collectively stated that the *N*-Gram robot seemed to learn a lot faster, as well as providing a much more difficult challenge. One tester stated that the *N*-Gram robot behaved like it knew what they were going to do next, as well as delivering the feeling of playing against an experienced human player. This is a great prospect for games with NPC opponents or allies, as human-like AI characters can help to increase immersion and the player's enjoyment of the game.

Interesting

In terms of how interesting testers found each method, *N*-Grams again won out but only slightly, with a value of 4.16 on average out of 5 compared to *Q*-Learning's 3.75 average value. A high value for how interesting testers found both methods is beneficial, because it is important for players to take interest in AI agents in games as they are generally what help the player engage with game play and story elements of a game. One tester stated that *Q*-Learning still acted unpredictably and exciting even after it had learned, which helped to keep the fight interesting. Another explained that the *N*-Gram agent was a lot more interesting because of the greater challenge it provided as well as how fast and efficiently it learned.

Enjoyable

The testers' opinion on how enjoyable a learning method was is of course a personal preference when it comes to playing games, however it is important to look at a wide range of players with different tastes to understand how the implementation of learning could affect them. On average, *Q*-Learning had an average value of 3.75 out of 5 for how enjoyable they found fighting the robot, whereas there was a small increase in the average value for *N*-Grams which had a value of 3.83 out of 5. The testers were additionally asked to identify which method they found most enjoyable and why, and there was no particular preference shown.

Learning in Games

In the final section of the questionnaire, testers were asked general questions on their opinion of learning AI agents in games to determine if this type of AI would appeal to them in the future.

Firstly, testers were asked whether they believed that the ability to learn made the AI agents in the test game more realistic and reactive in comparison to agents in other games they had experienced. Every tester responded positively to this question, with most citing games wherein the AI agent's behaviour can be quite illogical and easy to trick or exploit. Players mentioned these games and how agents that learn from the player would avoid the problem of repetitive, boring or exploitable AI by instead being unpredictable and surprising the more it learns about the player. Testers noted that in multiplayer games, humans do learn from their enemies or their allies and base their own play style on what they have learned for their own benefit. Therefore, AI agents that too can learn would be able to exhibit this realistic, human like behaviour. One player additionally stated that a learning AI could help to improve the difficulty level of a game substantially by tailoring it to individual players to improve their game play experience. This illustrates the many ways that games could improve player's interactions with AI agents or systems.

In order to get an impression of what players are looking for in future games testers were asked if having learning AI agents in these games would appeal to them and the overwhelming response was 100% yes.

Quantitative Results

Processing Speed

In order to test and compare the processing time of the learning methods implemented, the evaluation and optimisation tool within Unity, the Unity Profiler, was used. To gain an overall idea on how the processing time for each learning method compared, a sample of ten processing times for each method were recorded and then averaged in order to find the mean processing time required to carry out learning. Figure 6 shows the results.

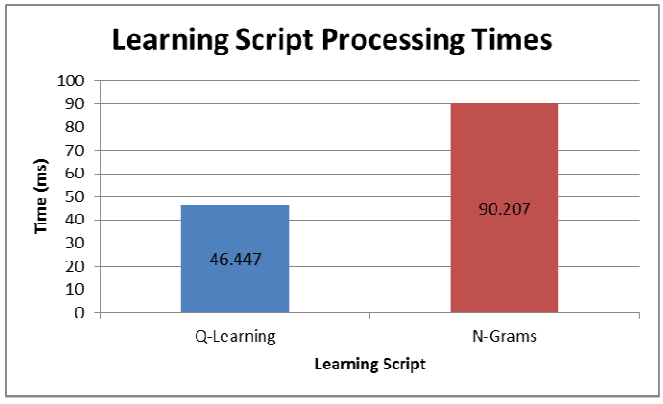

Figure 6: Bar Chart Showing the Comparison Between Processing Times

As evident in the above chart, *Q*-Learning was faster than *N*-Gram by about 45 milliseconds, and while this is a small difference this could have a much larger knock on effect in other games if the methods were used to control more agents, or to learn a wider range of knowledge. In comparison to *N*-Grams, *Q*-Learning has a relatively smaller amount of variables to search through in order to make decisions which could contribute to the reason why *Q*-Learning is faster. This is because during the *N*-gram based learning, the script has to check through every listed sequence that has happened and every action in that sequence in order to find matches for predictions. This list grows as the game goes on. However, for *Q*-Learning the script is only required to search through a list of 8 potential actions based on the state which is given to them by the AI robot script. This would reduce the processing time as there are less values or variables to search through to find the optimal action.

Of course, in future implementations each method's effect on the performance of a game could be improved further by using optimisation techniques such as threading. Nevertheless, it is always important and preferred that the efficiency of AI methods implemented in games are as fast as they can be so that they do not have a negative effect on the game's performance.

Accuracy and Error

To investigate the effectiveness of the methods, the amount of errors that were made were recorded over time in order to show if learning was taking place. Ideally, the error should decrease as the AI agent experiences more events during the game as this would display how the method stores more accurate knowledge as time goes on.

The errors of the *N*-Gram based system and the *Q*-Learning method were compared with each other in order to determine which has the higher rate of success when choosing its actions to counter the player. To determine the error for each method the percentage of incorrect decisions the AI agent made during the fight was calculated. For the *Q*-Learning method an error was when the agent made a 'wrong,' decision by selecting an action that would lead to a negative reward. For the *N*-Gram method, the error was based on whether the predicted action matched the actual action used by the player. To compare the error percentages, the error was recorded over 25 player moves (game events) to indicate how well the agent learned. Figure 7 displays the results for this test.

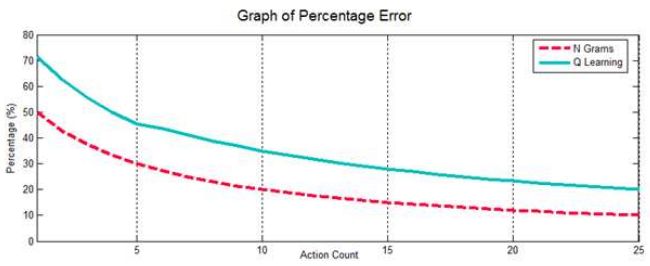

Figure 7: Comparison of Percentage Error

As shown clearly above, both methods have a similar learning rate at the start despite the *Q*-Learning method beginning at a higher error percentage. However as time goes on and the learning rate slows, the *Q*-Learning rate levels out at a higher error percentage than the *N*-Gram based system by becoming flatter at around 20%, whereas N-grams achieves this at 10%. In addition, throughout the graph the *N*-Gram method always has a smaller error percentage which demonstrates that the method is slightly more efficient at learning than the *Q*-Learning method as it tends to make less mistakes as time goes on. This supports the comments of the testers of which a majority stated that the *N*-Gram agent felt more intelligent and reactive compared to the *Q*-Learning agent. However, because the game is a fighting game that requires clear input and reactive output constantly this result could simply be indicative of *N*-Gram prediction being more suited for this style of input and learning. *Q*-Learning takes slightly longer, however this could be beneficial for different games that require a more subtle or natural sense of learning. Moreover, both methods show a decrease in the error percentage as they experience more events which shows that they both successfully learn and improve the AI agent's behaviour throughout the game. This in turn illustrates how both methods would be beneficial when implemented in games to clearly display the intelligence of the agents and increase their reactivity.

DISCUSSION

Real-time Concerns

One of the main problems that the project looked to explore was how agents could learn directly from the player whilst the game is played in real time without having a negative effect on the performance of the game. The project found that players did notice the AI agent learning from their actions during run time and they found this to be interesting and enjoyable, illustrating that the *N*-Gram and *Q*-Learning methods were both effective in facilitating fast learning. By exhibiting behaviour based on the knowledge that the AI agent had learnt from the player in the short, 45 second game, it is clear that machine learning methods are suitable for adapting agents as the game is being played. However, it is incredibly important to carefully plan what the AI agent is able to learn, as well as how the agent will change its behaviour based upon this information. It is much safer to utilise machine learning in games to select the decisions that the agent should be capable of making rather than giving the learning methods free control over all behaviour of the AI agent. In this way, the game play is still unpredictable and exciting without causing unstable or illogical behaviour.

It was found that *Q*-Learning was the most efficient out of the two tested methods as shown above, however it was also the method that took longest to learn in comparison with the *N*-Gram system. The *N*-Gram system required a longer processing time than *Q*-Learning, however the majority of player testers preferred this method as it felt the most realistic and seemed to learn faster. This is also reflected in Figure 7 wherein the *N*-Gram agent had a lower error percentage throughout, illustrating that it was more successful in predicting the players moves than *Q*-Learning was in choosing the most rewarding move to make.

The Player Experience

In order for the application of machine learning methods to be beneficial and a worthwhile innovation in games, the AI had to enhance the realism and reactivity of the agent towards players, as well as exhibit human-like intelligence. AI within games should satisfy one goal, which is to help to 'create a compelling experience for the player (Dill 2011).' The qualitative testing found that 100% of testers would welcome real time learning agents in games in the future, citing reasons such as how the ability to learn improved the

realism and challenge of the test game as well as the reactivity of agents. Many testers explained that they believed innovation in AI is the future for games, and learning is just one of many aspects that could change and enhance the player's engagement with games. This illustrates the relevance of this project and the research undertaken, because in the current games industry environment developers are constantly looking for new ways to entice and provide fun for gamers. The qualitative information gathered in the project is strong evidence in support of the statement that machine learning can be applied to games to improve the realism and reactivity of AI agents.

An area of the results that was unexpected was how some testers felt that when the AI agents behaved too intelligently, this actually negatively affected their game play experience. These testers signified how the speed of the AI agents learning actually changed how much fun or frustration they got from the game, in addition to how much chance they felt they had to beat the AI robot. *N*-Grams was found to be the most reactive method as well as the fastest learning, however many players stated they disliked this behaviour as they felt their efforts were futile in fighting it and this removed the element of fun. On reflection, it is important that the AI agent's intelligence is balanced so as to still provide unpredictable behaviour, but should not be too intelligent or reactive that they can anticipate every move of the player. This frustrates players as they feel there is no point in playing the game if there is no chance of winning. Testers suggested that a larger element of randomness could improve the learning methods as it would make their behaviour seem slightly more natural. This is because players often make mistakes or switch up their tactics while playing games, and this not only would benefit the player in terms of showing them there is a higher chance for them to win but it would also benefit the agent by giving them human-like intelligence, along with human-like fallibility.

Future Work

While the learning methods implemented in this project focused on learning reactions to the player through game play, machine learning could similarly enhance many other areas of games. For example, area or terrain generation could be autonomously created by utilising machine learning and could give randomised locations for the player in each play through. In addition, areas such as narrative, graphics and networking may benefit from machine learning (Graepel and Herbrich 2008). Experimentation into using machine learning in different ways could lead to more optimal methods of creating content for games.

As discussed above players noted that the difficulty of the game seemed to depend on how fast the AI agent could learn to counter the player. In this sense, a game that utilised learning agents as enemies could adjust their learning rate depending on the difficulty the player prefers or simply to ensure that the game remains a challenge even as the player improves. This would be a useful and interesting way to adjust the difficulty of a game instead of simply changing health values and damage values to make games harder. As the game becomes more difficult, the AI agent could learn different types of information about the player that were not previously available to them, and this would keep the game play challenging as well as unique.

CONCLUSION

The findings of the project have shown that integrating AI learning in a game is a worthwhile task for developers, as it greatly enhances the behaviour of AI agents as well as the player's engagement with the game. Players found that the learning ability of agents led to exciting, unpredictable and realistic behaviour that enhanced their immersion and enjoyment of the game. Yet, documentation and tools on the subject of machine learning in relation to games are lacking or often focused on offline learning rather than online learning. It is likely that machine learning in games would be a more common occurrence if game engines created tools to allow developers to easily utilise learning in games. In addition, documentation on how machine learning methods could be applied to games that focuses on the type of learning the method utilises and to which game genres each is best suited would be useful. The benefit being that it would help to increase developers understanding of online learning and encourage them to investigate using it, as right now many developers deem it too great a risk. Nevertheless, taking such a risk could result in a ground-breaking game with revolutionary game play.

For more details on this investigation, including experimenting with ANN, please refer to Bennett 2016.

REFERENCES

AI Horizon [no date] *Machine Learning, Part 1: Supervised and Unsupervised* Learning [on line]. Available from: http://www.aihorizon.com/essays/generalai/supervised_unsupervised_machine_learning.htm.

Bennett, C. 2016. *A comparison of Real Time Machine Learning Techniques to Indentify Those That Enhance the Adaptibility of Game AI Agents*. Dissertation, Abertay University.

Bourg, D.M. and G. Seemann. 2004a. "Introduction to Game AI". In *AI for Game Developers 2004*. O'Reilly Media Inc., Sebastopol, CA, 1-2.

Bourg, D.M. and G. Seemann. 2004b. "Preface". In *AI for Game Developers 2004*. O'Reilly Media Inc., Sebastopol, CA, 1-2.

Capcom 1987. *Street Fighter*. Capcom, Japan [Arcade Game].

DeWolf, T. 2012. *Reinforcement Learning Part 1: Q-Learning and Exploration* [online]. Available from: https://studywolf.wordpress.com/2012/11/25/reinforcement-learning-q-learning-and-exploration/.

Dill, K. 2011. "What is Game AI?". In *Game AI pro: Collected Wisdom of Game AI Professionals2011*, S. Rabin (Ed.). CRC Press, Boca Raton, FL, 3-8.

Graepel, T. and R. Herbrich. 2008. *Video Games and Artificial Intelligence* [online]. Available from: http://research.microsoft.com/en-us/projects/ijcaiigames/.

Lewis, T. 2015. *Google's Artificial Imntelligence Can Probably Beat You At Video Games* [online]. Available from: http://www.livescience.com/49947-google-ai-plays-videogames.html.

Mathworks [no date]. *Machine Learning Technique for Building Predictive Models from Known Input and Response Data* [online]. Available from: http://www.mathworks.com/discovery/supervised-learning.html?s_tid=gen_loc_drop.

Midway Games 1992. *Mortal Kombat*. Midway Games [Arcade Game].

Millington, I. and J. Funge. 2009. "Learning". In *Artificial Intelligence for Games*. Morgan Kaufmann, Burlington, MA, 597-598.

NERO Team [no date]. *NERO: NeuroEvolving Robot Operatives* [online]. Available from: http://nerogame.org/.

Poole, D. 2010. *9.5.3.Value Iteration* [online]. Available from: http://artint.info/html/ArtInt_227.html.

Poole, D. and A. Mackworth. [no date] *11.3.3. Q-Learning* [online]. Available from: http://artint.info/html/Artint.265.html.

Schwab, B. 2009. “Neural Networks”. In *AI Game Engine Programming 2009*, Course Technology, Boston, MA, 558-570.

Spronck, P. I. Sprinkhuizen and E. Postma. 2003. *Online Adaptation of Game Opponent AI in Simulation and in Practice*. Maastricht University, Maastricht.

Stanley, K.O., B.D. Bryant and R. Miikkulainen. 2005. “Real-Time Neuroevolution in the NERO Video Game”. *IEEE Transactions on Evolutionary Computation* 9, No.6 (Dec), 1-2.

Tucci, A. 2014. *AI for Games Seminar: N-Grams Prediction + Intro to Bayes Inference* [online]. Available from: http://www.slideshare.net/andreatucci91/ai-for-games-seminar-ngrams-prediction-intro-to-bayes.

Vasquez II, J. 2011. “Implementing *N*-Grams for Player Prediction, Procedural Generation, and Stylized AI”. In *Game AI pro: Collected Wisdom of Game AI Professionals2011*, S. Rabin (Ed.). CRC Press, Boca Raton, FL, 567-580.

Watkins, C.J.C.H. and P. Dayan. 1992. *Technical Note: Q-Learning*. Kluwer Academic Publishers, Boston, MA.

Whiteson, S.A. 2007. *Adaptive Representations of Reinforcement Learning*. The University of Texas at Austin.

AUTHOR BIOGRAPHIES

DAVID KING is a lecturer in Mathematics and Artificial Intelligence at Abertay University teaching on the Computer Games Technology and Computer Games Application Development Programmes. Email: d.king@abertay.ac.uk.

CASSIE BENNETT graduated from Abertay University in July 2016 with a BSc (Hons) First Class in Computer Games Technology..
Email: CassBennettDev@gmail.com.